Dual-V_{DD} and Input Reordering for Reduced Delay and Subthreshold Leakage in Pass Transistor Logic

Jeff Brantley and Sam Ridenour
ECE 6332 – Fall 2010
University of Virginia
<jb7fx,sar7f>@virginia.edu

ABSTRACT

Pass transistor logic (PTL) typically suffers from reduced output swing and correspondingly slow delay. We examine the use of dual-V_{DD} supply to separately drive PTL gates at a higher voltage than the source-drain terminal inputs, achieving 2-3 orders of magnitude delay reduction in a PTL multiplexer. Furthermore, we describe an input reordering technique for reduction of "sneak" leakage in a PTL mux, such as would be used in an FPGA lookup table (LUT). With this technique, we achieve a 19% reduction in PTL sneak leakage and an overall leakage reduction of 2.5%.

1. INTRODUCTION

As transistor sizes have continually scaled down, leakage current has contributed more and more to overall energy consumption, motivating research into leakage reduction techniques, including subthreshold operation for ultra-low power applications. One potentially interesting technique is the use of pass-transistor logic (PTL), in which transistors are interconnected only by their source-drain terminals rather than leaving direct leakage paths through pull-up and pull-down networks positioned between the supply rails. This approach can be accomplished with transmission gates, or better yet, with only nMOS transistors for a reduction in area, with the consequence of a reduced output voltage swing, giving rise to increased delay and thus exacerbating the effects of subthreshold operation. However, it has been observed that driving the gate terminals of a PTL network at a higher-than-V_{DD} level can improve this delay. Thus, in order to address the problem of reducing leakage while taking advantage of the area reduction offered by nMOS PTL and yet still limiting the associated increase in delay, we propose the use of dual-V_{DD} rails for separately driving gate and source-drain terminals of a PTL network.

A potential caveat to working with PTL networks is the possibility for "sneak" leakage paths between source-drain terminal inputs. The existence of such paths is dependent upon the combination of inputs supplied to create a potential (i.e. a logic '1' on one and a logic '0' on another, connected via a transistor in cutoff). We observe that, in the context of a programmable lookup table (LUT) implemented with a PTL mux, the select lines and input lines of the mux may be reordered together to manipulate the pattern of such sneak leakage paths to a lower-leakage state. We apply an algorithm based on this concept for leakage reduction in a 4-bit LUT.

The contributions of this work are, namely:

- the application and evaluation of dual-V_{DD} supplies for raised gate terminal voltage in a PTL mux for improved energy-delay characteristic, and
- a novel leakage reduction technique for FPGA LUTs implementable at synthesis/place-and-route time with no effect on hardware design.

The remainder of this paper is organized as follows. Section 2 highlights related work in the use of dual-V_{DD} and leakage reduction based on input pattern manipulation. Section 3 discusses our dual-V_{DD} and input reordering approaches. Section 4 presents our experimental results, and finally Section 5 concludes our evaluation of the proposed techniques.

2. RELATED WORK

Previous research has proposed the use of dual-V_{DD} supplies, for example in FPGAs. Li et al. propose alternating rows of high- and low-voltage blocks in an FPGA, with placement being chosen for best performance, i.e. critical-path blocks on the higher supply [5]. Gayasen et al. propose a similar scheme with the supply choice set programmatically [3]. Both schemes segment the supply per logic block so that some blocks may be lower-power. In contrast, Ryan et al. propose a dual-V_{DD} scheme in which the boundary is at the FPGA interconnect configuration; the interconnect signals through switch block passgates are low-swing while the gate terminals are driven by high-V_{DD} inputs statically configured in SRAM bitcells [6]. This approach offers a 14X speedup and 22X reduction in energy in a subthreshold FPGA's interconnect. However, the method is not directly evaluated in the lookup tables (LUTs) or in mux-based routers inside a single common logic block (CLB).

In [1], Alarcón et al. explore a PTL LUT-like structure "programmed" at design-time by selective wiring. One stated advantage of this circuit is to reduce the existence of "sneak" leakage paths typically found in PTL, in which leakage current can pass from one source-drain input to another. However, this circuit comes with hidden area, energy, and delay costs required for a clocked pre-discharge cycle prior to each evaluation. Anderson et al. observe that leakage in a PTL programmable LUT is reduced as more of the source-drain inputs are logic '1's, and forces a majority of '1's as necessary by inverting all bits, thus complementing the output and requiring a rearrangement of bits in any downstream LUTs [2]. Hassan et al. proposes reordering the select lines and input lines together into formations that minimize leakage, removing potentials by causing more bits across leakage paths to be equal [4][1].

3. APPROACH

While PTL can be used to implement arbitrary logic functions, we select a PTL mux as our case study for evaluation. The structure

[1] This is the same approach we have pursued in this work; we discovered it only after completing our own evaluation.

of the mux used is shown in Figure 1 with the select lines driving the gates of transistors in order to choose among the available inputs. The mux pictured is a 4:1 (2 stage) mux, but we conduct all simulations on a 16:1 (4 stage) mux.

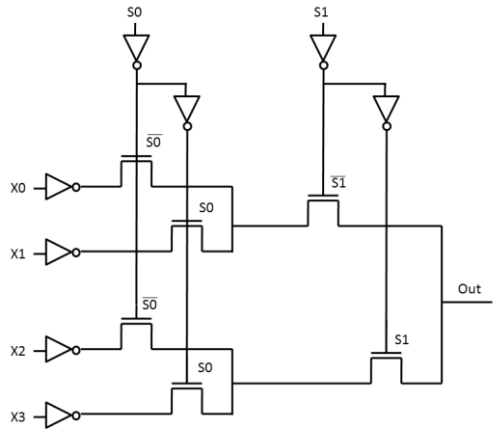

Figure 1. Pass transistor multiplexer (4:1). Select lines and complements choose between available inputs at each stage.

3.1 Dual-V_{DD} Pass Transistor Logic

In the base case of a single V_{DD} applied to the mux, the output has a reduced swing due to a Vt drop across the transistors, as shown in Figure 2. However, we can apply dual-V_{DD} to the mux by raising the voltage of the select lines. By driving the gates at a higher voltage, the delay through the mux is reduced and the effect of the Vt drop across the transistors is reduced, allowing the output to have a greater swing, as indicated in Figure 2.

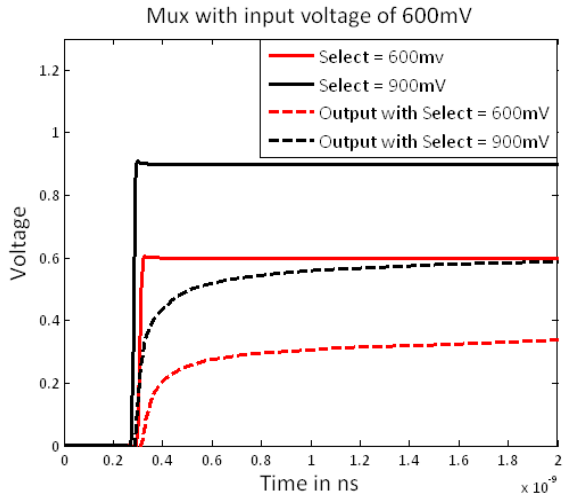

Figure 2. Output of 16:1 mux in single-V_{DD} and dual-V_{DD} cases, with V_{DDL}=0.6V. Slow-rising output is restored to full-rail by applying V_{DDH}=0.9V to transistor gate terminals.

Due to the reduced output swing in single V_{DD}, a keeper is typically used to restore the output to full swing. However, as V_{DD} is lowered, the Vt drop across the transistor becomes a greater proportion of the output voltage. This Vt drop can cause the output voltage to very slowly reach half of V_{DD}, as shown in Figure 2, or not at all, rendering a keeper ineffective. In these cases, a more sophisticated circuit is needed, such as a Schmitt trigger or sense amplifier. However, in order to simplify our evaluation and limit the influencing variables, we exclude the effects of a level-restoring device in our analysis. Consequently, when measuring delay, our metric is defined simply as 50% of the input swing to 50% of the output swing.

3.2 Input Reordering for Leakage Reduction

In PTL circuits, some of the inputs are at source-drain terminals of transistors, rather than only the gates. Because of this, "sneak" leakage current paths can exist between multiple inputs driving the same PTL network, depending on the structure and temporal values of said inputs. Figure 3 illustrates this concept with a 4:1 PTL mux. Inputs X0 and X1 are set to opposite values, creating a leakage path through transistor M2, while X2 and X3 similarly create a leakage current through M4. On the other hand, inputs X0 and X2 are both logical '1', and so there is effectively no leakage current through M6.

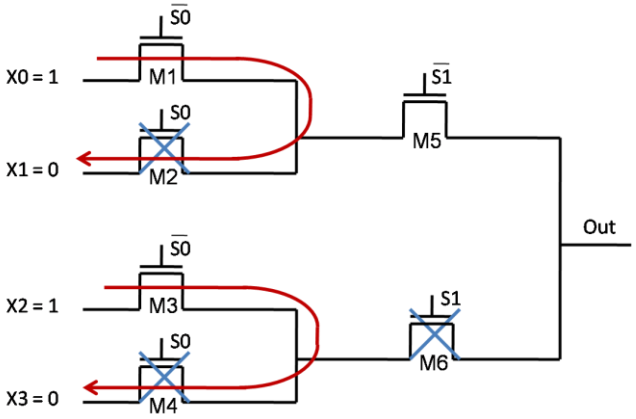

Figure 3. Sneak leakage between inputs to a PTL mux.

This dependence of sneak leakage on the state of the inputs can be exploited to reduce the average leakage by rearranging the inputs into a lower-leakage state. We observe that, for a multiplexer, it is possible to reorder the select lines and, correspondingly, the input lines, while maintaining the logical correctness of the output. For example, in Figure 3, swapping the order of select lines S0 and S1 would require swapping the locations of inputs X1 and X2 in order to maintain the same logic function. Doing so would change the input pattern X3:0 from '0101' to '0011', thus eliminating the leakage currents through M2 and M4, but introducing a single leakage path through M6, for a net reduction in leakage current.

We developed a two-part algorithm based on this principle with an FPGA lookup table (LUT) as our target, and thus the X inputs are fixed values stored in SRAM bitcells. The first part computes a relative leakage score for a given SRAM vector programming and static probabilities of a logic '1' at each select line. The second part is an exhaustive search algorithm that tests the leakage score for every possible permutation of the select lines, and chooses one with a minimum score.

This approach to leakage reduction can be applied at the time of synthesis/place-and-route, requiring no changes in hardware design. The inputs may be reordered via flexible routing, if available, or by swapping the placement of LUT functions, but note that due to dependence among functions, it may not always

be possible to achieve a global optimum, and placement changes must be balanced with potential impact on performance.

4. RESULTS

4.1 Dual-V_{DD} Pass Transistor Logic

We first explored the PTL mux in the context of a LUT in an FPGA, i.e. the 16 mux inputs are set at programming and all activity is confined to the select lines. For a number of starting V_{DD}s (denoted as V_{DDL}), we hold this voltage constant and sweep the select line V_{DD} (denoted as V_{DDH}) until (a) the delay stops decreasing or (b) V_{DDH} reaches 1.1V.

Figure 4 shows the resulting family of energy-delay curves, one for each choice of V_{DDL}, along with the single V_{DD} base case. As the select line V_{DD} is raised, there is a decrease in delay and energy compared to the base case. Eventually, this leads to diminishing returns on the delay improvement, while the energy of the inverters on the high V_{DD} supply begin to dominate overall energy consumption, and thus the dual V_{DD} begins to perform worse than the base case.

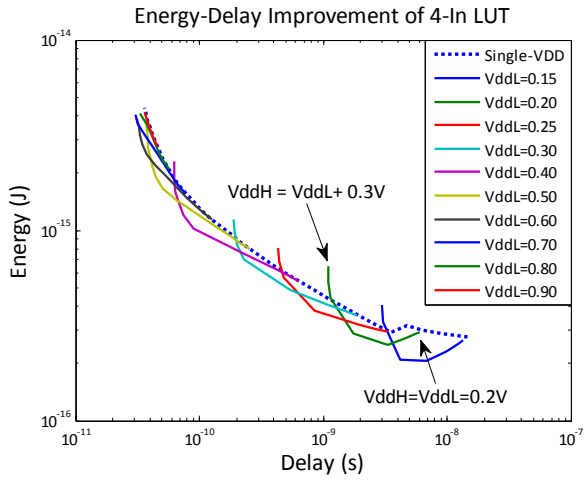

Figure 4. Energy-delay plot for mux-as-LUT for a range of V_{DDL} choices. For each case, V_{DDH} is swept up from V_{DDL} until diminishing delay improvements are reached.

Next, we examine the case of the PTL mux as a router, in which all activity occurs on the input lines and the select lines are fixed. Again, V_{DDH} is swept for each choice of V_{DDL}. The resulting energy-delay curves are shown in Figure 5. Compared to the base case, dual-V_{DD} achieves orders of magnitude reduction in delay for a relatively low energy increase. These results are significantly better the LUT case, due to the fact that the driving inverters that are switching are now on the low supply rail and the select line are no longer on the critical path. In fact, except for the case where V_{DDL}=0.15V, delay continues decreasing as V_{DDH} is swept all the way up to 1.1V.

The previous two mux contexts represent the two extreme cases of activity ratio between the input lines and select lines. Figure 6 and Figure 7 show the continuum of delay and energy as a function of this activity ratio as it varies between the two extremes, for the case where V_{DDL}=0.2V. The left edge of the graph corresponds to the LUT case, whereas the right edge represents the router case. Each is labeled with a ΔV corresponding to the difference between V_{DDL} and V_{DDH}.

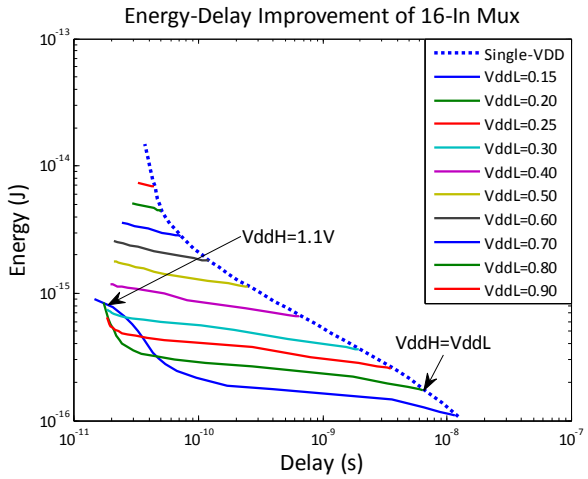

Figure 5. Energy-delay plots for mux-as-router for a range of V_{DDL} choices. For each case, V_{DDH} is swept from V_{DDL} to 1.1V, except for the V_{DDL}=0.15V case, where V_{DDH} stops at 1.0V.

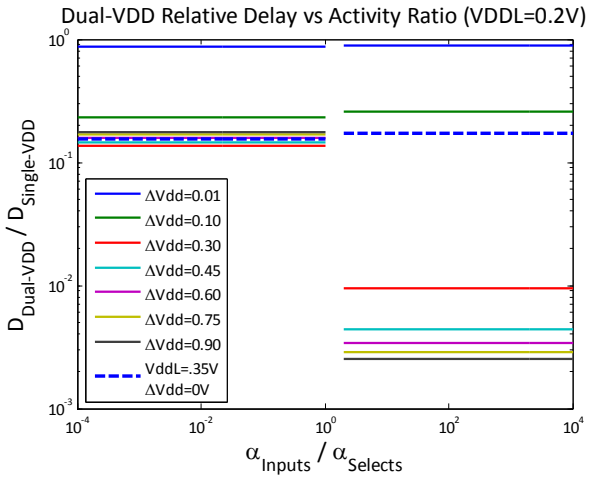

Figure 6. Delay through 16:1 mux as a function of activity ratio between input and select lines, with V_{DDL}=0.2V and a range of $\Delta V = V_{DDH} - V_{DDL}$.

As Figure 6 shows, the delay is unaffected by activity ratio, except at some point where activity is more common in the select lines, such that the critical path is redefined as inputs-to-output rather than selects-to-output, which is slower. We have marked this boundary where the $\alpha_{Inputs} / \alpha_{Selects}$ ratio moves from 1 to 2, but the validity of this depends on the final circuit and timing.

While the delay remains essentially constant, with one discontinuity point, the energy gradually decreases, especially in the neighborhood where activity ratio is between 1/10 and 10/1, except for the lowest choices of ΔV. Again, this is due to the shift of switching power from the ever-increasing high rail to the low rail. It is worth noting that other design points, such as the single-V_{DD} case where V_{DDL}=0.35V, as shown in Figure 7 can require either more or less energy for similar delay as certain ΔVs

depending on the exact activity ratio, although in the case shown, there is no contest once inputs-to-output becomes the critical path.

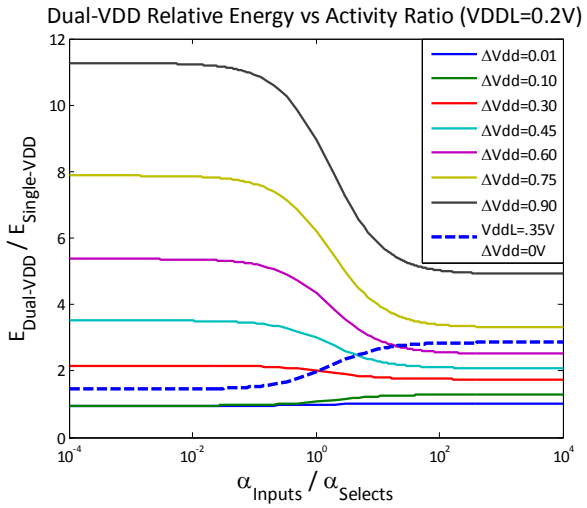

Figure 7. Energy per operation for 16:1 mux and drivers vs. activity ratio, with V_{DDL}=0.2V and a range of ΔV.

4.2 Input Reordering for Leakage Reduction

We evaluated the input reordering algorithm of Sec. 3.2 on a 16:1 nMOS PTL mux on the 45nm FreePDK PTM. All terminals were sourced directly by rails (i.e. no static CMOS drivers included), and total leakage was measured for every SRAM vector, with select lines set to logic '0000' (the data for all other select line values can be inferred due to symmetry in the mux). The average leakages before and after our optimization is applied were computed for 30 select line S3:0 static probability vectors (randomly drawn from a uniform distribution). Figure 8 demonstrates the general pattern of leakage with respect to all 65,536 possible SRAM vectors, as well as the mean leakage before and after optimization. The mean is reduced from about 62 nA to 50nA. This equates to a mean leakage reduction of 19.3%, with standard deviation 6.5%.

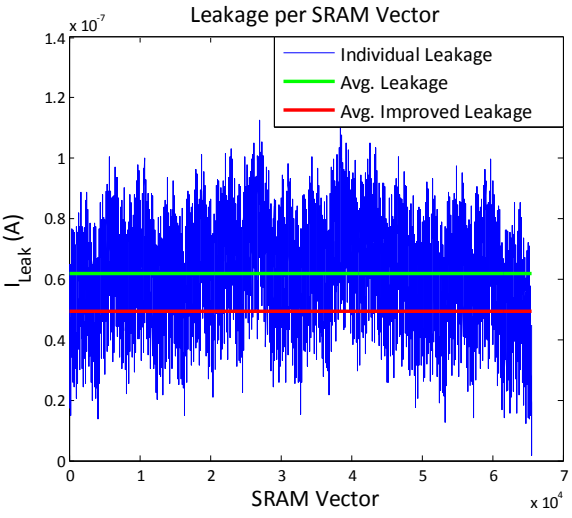

Figure 8. Average leakage for each possible 16-bit SRAM vector, showing mean before and after optimization.

However, when the same circuit is simulated in the form of Figure 1, with inverters driving select and input lines, the leakage pattern with respect to SRAM value changes shape (see Figure 9), now corresponding directly to the number of logic '1's vs. logic '0's due to a difference in leakage between in pMOS and nMOS *in the drivers*. That is, leakage in the drivers dominates over sneak leakage, and so the percentage improvement falls to a mean of 2.5% with standard deviation 0.9%.

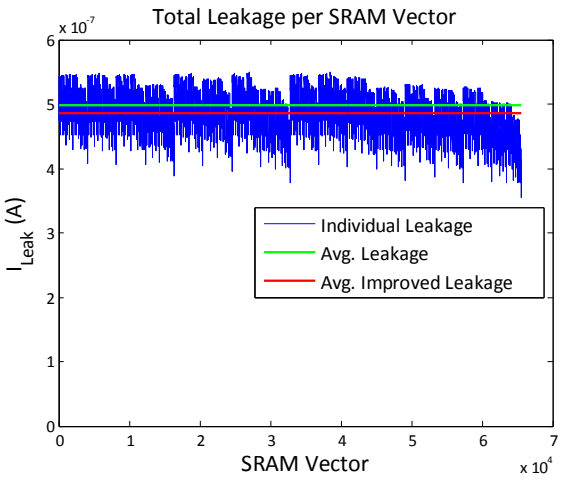

Figure 9. Average leakage including drivers for all SRAM values, showing mean before and after optimization.

5. CONCLUSIONS

Our results demonstrate that dual-V_{DD} split between the source-drain and gate terminal regions of a mux can offer substantial reduction of delay for minimal energy cost, especially if the dominant activity occurs at the input lines rather than the select lines. In the case studies presented, the technique gives a small improvement to a LUT, but 2-3 orders of magnitude reduction in delay for a mux router with less than an order increase in energy per operation. In general, the energy cost for delay improvement at a given dual-V_{DD} operating point decreases as the ratio of activity in the input lines vs. select lines increases, but the exact delay improvement can increase substantially at the ratio for which the critical path can be effectively redefined as inputs-to-output, rather than select-lines-to-output.

It should be noted that, while the results presented here look promising, there are some limitations. If PTL is used in a topology in which outputs eventually drive other gate terminal inputs (such as cascaded LUTs in an FPGA), some step-up is required between V_{DD} regions, imposing area, delay, and energy costs. Moreover, routing dual-V_{DD} will require additional area, and energy overhead due to dissipation in the V_{DD} routing will increase as V_{DDH} is increased, but we have not examined this effect here.

Finally, we show that input reordering can introduce significant leakage reduction (19.3±6.5%) for a mux in isolation (i.e. leakage due only to PTL sneak paths). Yet, in a test bench including static inverter drivers, driver leakage dominates, diminishing the leakage reduction to 2.5±0.9%. However, this case used all low-threshold devices, and a dual-threshold process in which the inputs (i.e. LUT SRAM) region is high-threshold could bring the leakage reduction percentage to some point in between.

Interestingly, we discovered this same method being used in [4], and the authors of that work report a 50% savings in overall leakage, which would seem impossible when one studies Figure 9 – that is, the swing in leakage as a function of SRAM is much smaller than the total leakage.

6. REFERENCES

[1] Alarcón, L.P., Pierson M.D., and Rabaey, J.M. Exploring very low-energy logic: a case study. *Journal of Low Power Electronics, 3*(3), 223-233.

[2] Anderson, J., Najm, F., and Tuan T. Active leakage power optimization for FPGAs. In *Proc. of ISFPGA,* (Monterey, CA, 2004), ACM New York, NY, 33-41.

[3] Gayasen, A., Lee, K., Vijaykrishnan, N., Irwin, M.J., and Tuan, T. A dual-VDD low power FPGA architecture. In *Proc. Int. Conf. Field Programmable Logic and Applications*, (Antwer, Belgium, 2004), Springer, 145-157.

[4] Hassan, H., Anis, M., Elmasry, M. Input vector reordering for leakage power reduction in FPGAs. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 27(9). 1555-1564.

[5] Li, F. Lin, Y. He, L. and Cong, J. Low-power FPGA using pre-defined dual-Vdd/dual-Vt fabrics. In *Proc. 12th Int. Symp. Field Programmable Gate Arrays*, (Monterey, CA, 2004), ACM New York, NY, 42-50.

[6] Ryan, J.F., Calhoun, B.H. A sub-threshold FPGA with low-swing dual-Vdd interconnect in 90nm CMOS. In *IEEE Custom Integrated Circuits Conference*, (San Jose, CA, 2010), IEEE.